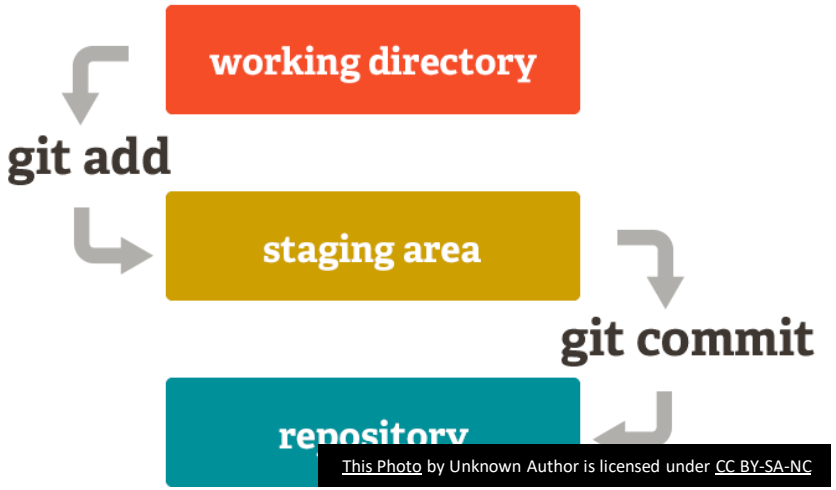
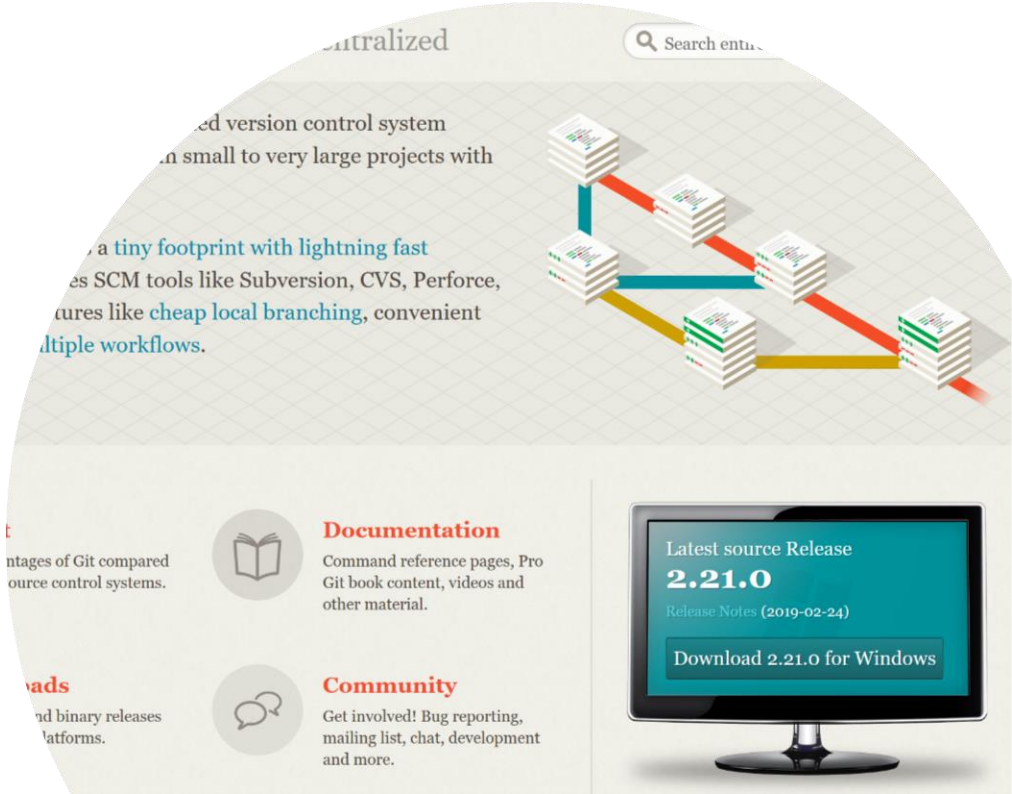


Git Basics

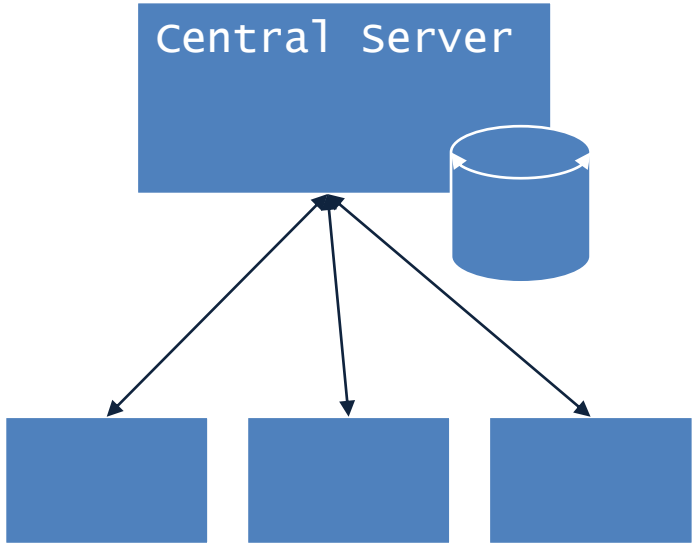
Frank Walsh

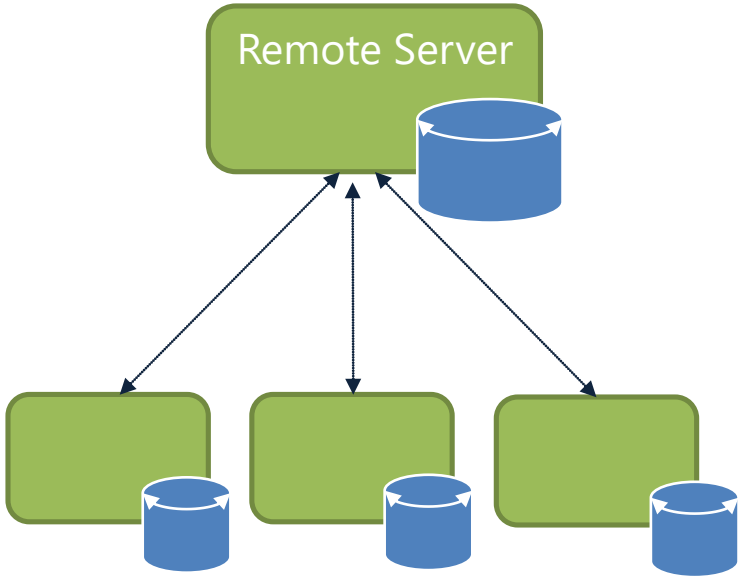
What's Git

- Distributed Version Control
- Directory Content Management
- Use it to keep track of your stuff
  - (and not just code…)
- I get confused sometimes!
  - (and I'm not the only one…)



Git is “distributed”





Initialising a Repo locally (command line)

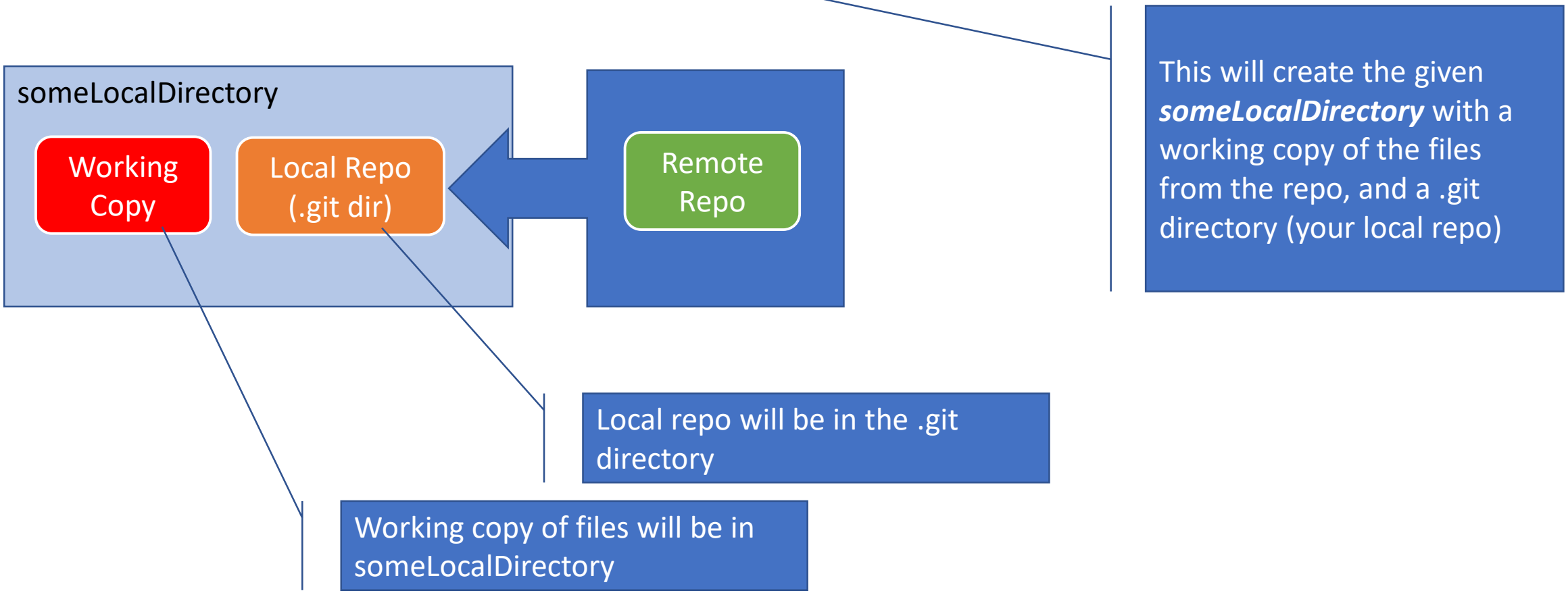
```
$ mkdir myproject
$ cd myproject
$ git init
Initialized empty Git repository in C:/Users/Frank/myproject/.git/
$ git config --global user.name “fxwalsh“
$ git config --global user.email fxwalsh@wit.com
$ touch README.md
$ git add .
$ git commit -m 'initial commit'
[master (root-commit) 7d738f4] initial commit
1 file changed, 1 insertion(+)
 create mode 100644 README.md
```

Initialises current directory as “working directory”. Creates .git directory – that’s your empty local repo.

Configures the default user name and email. Used to track repo actions

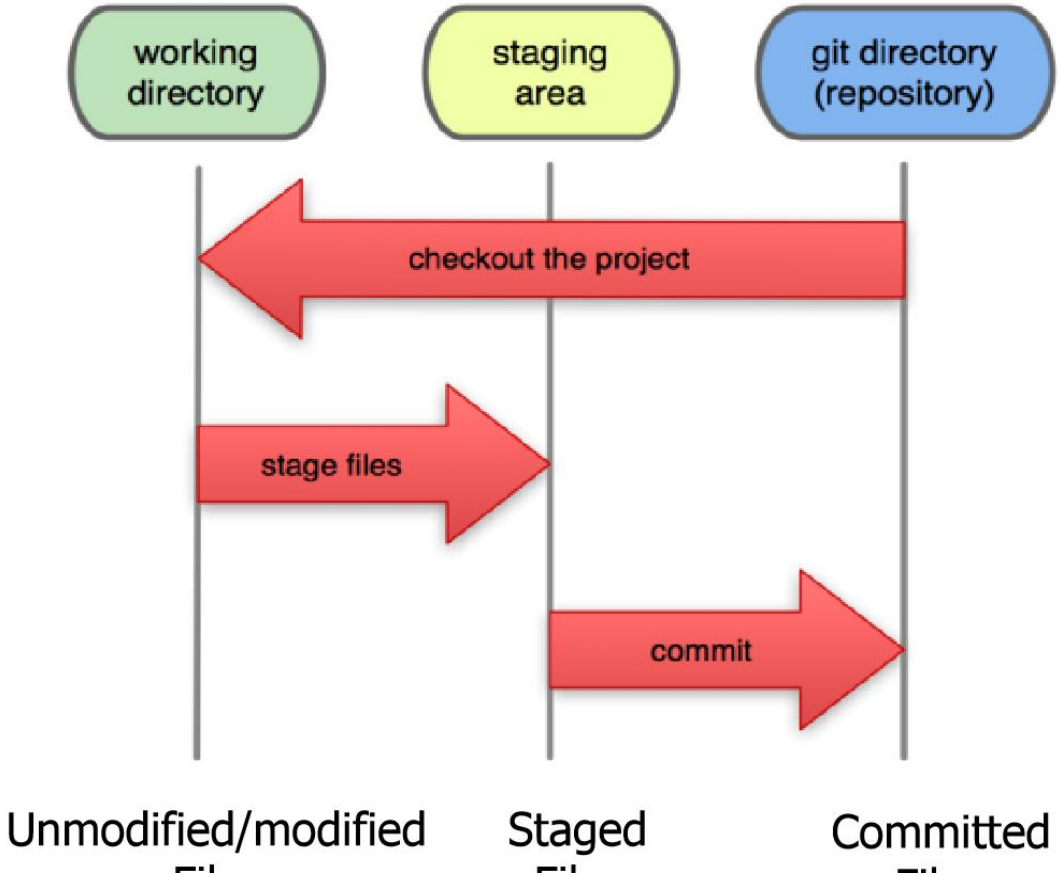
Alternative – clone a remote

- `git clone url someLocalDirectory`



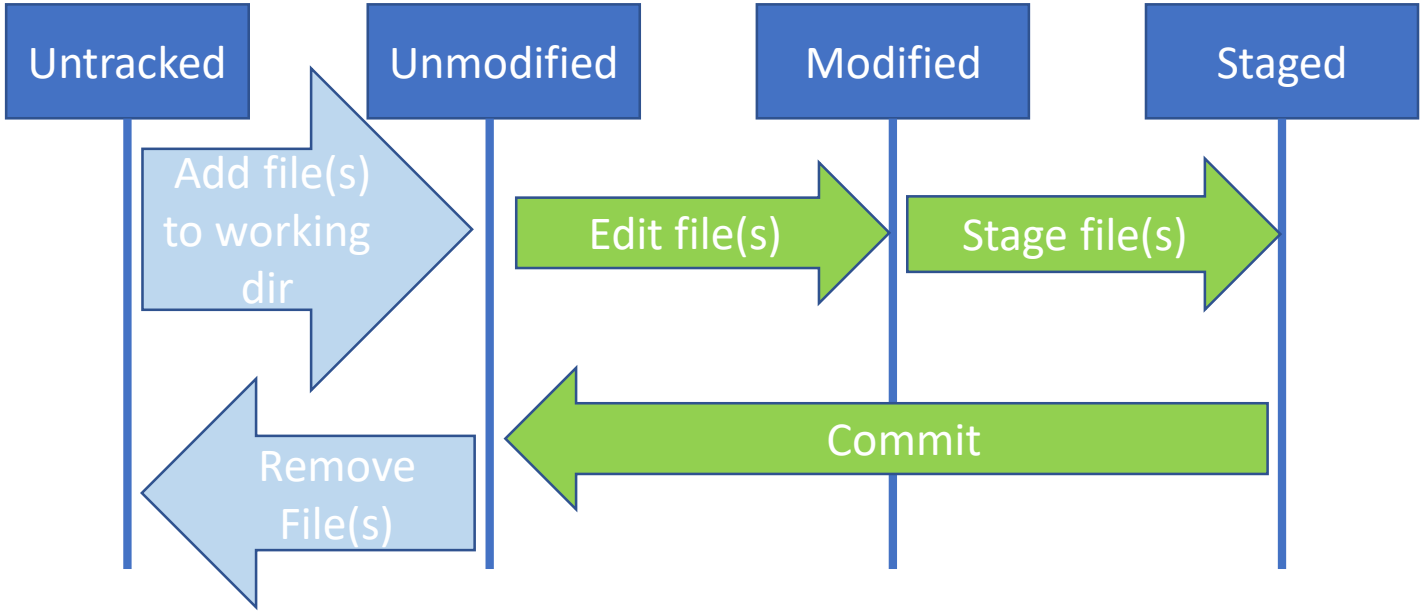
Local Git States

- Files can be
  - In your local repository
    - once committed
  - Checked out and/or modified
    - In your working directory(working copy)
  - Staged (using git add)
    - Ready to be committed
- A **Commit** saves a “snapshot” of (and only of) staged changes.
  - REMEMBER: JUST BECAUSE YOU SAVED A FILE DOESN’T MEAN GIT WILL INCLUDE IT IN THE REPO. IT MUST BE STAGED FIRST.

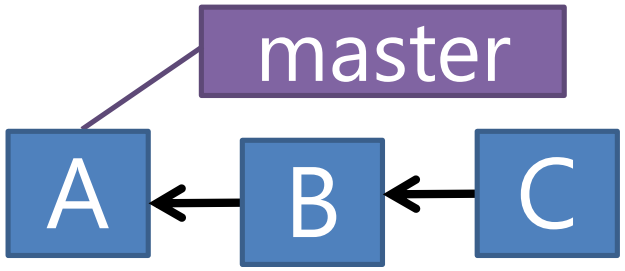


Typical Git Workflow

- Edit your files in the working directory
- Stage files once your happy with them(e.g. the script works!)
- Commit at significant junctures (e.g. functionality finished, embarking on significant code changes/refactoring, publishing work)



Multiple Commits

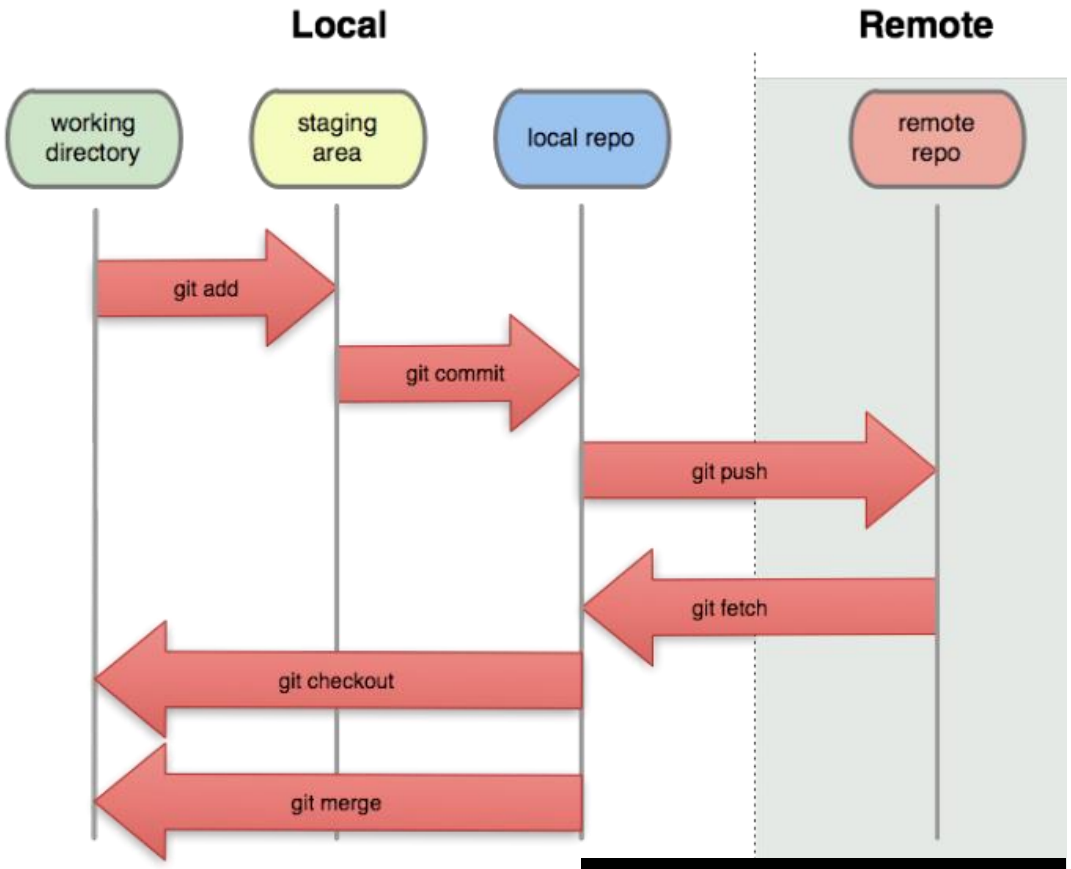


$ git add README.md
$ git commit –m “updated read me”
[master (root-commit) 7d738f4] ….
$ git add --all
$ git commit –m “updated read me again”
[master (root-commit) 627d8h4] ….

command	description
`git clone` ***`url [dir]`***	copy a Git repository so you can add to it
`git add` ***`file`***	adds file contents to the staging area
`git commit`	records a snapshot of the staging area
`git status`	view the status of your files in the working directory and staging area
`git diff`	shows diff of what is staged and what is modified but unstaged
`git help` *`[`****`command`****`]`*	get help info about a particular command
`git pull`	fetch from a remote repo and try to merge into the current branch
`git push`	push your new branches and data to a remote repository
others: `init, reset, branch, checkout, merge, log, tag`	

Remote Repositories

- Online storage providers of Git repositories include Github and Bitbucket.
  - You can create a remote repo there and push code to it.
  - Many dev teams used/collaborate with them in various modes
  - You can get free space for open source and private projects

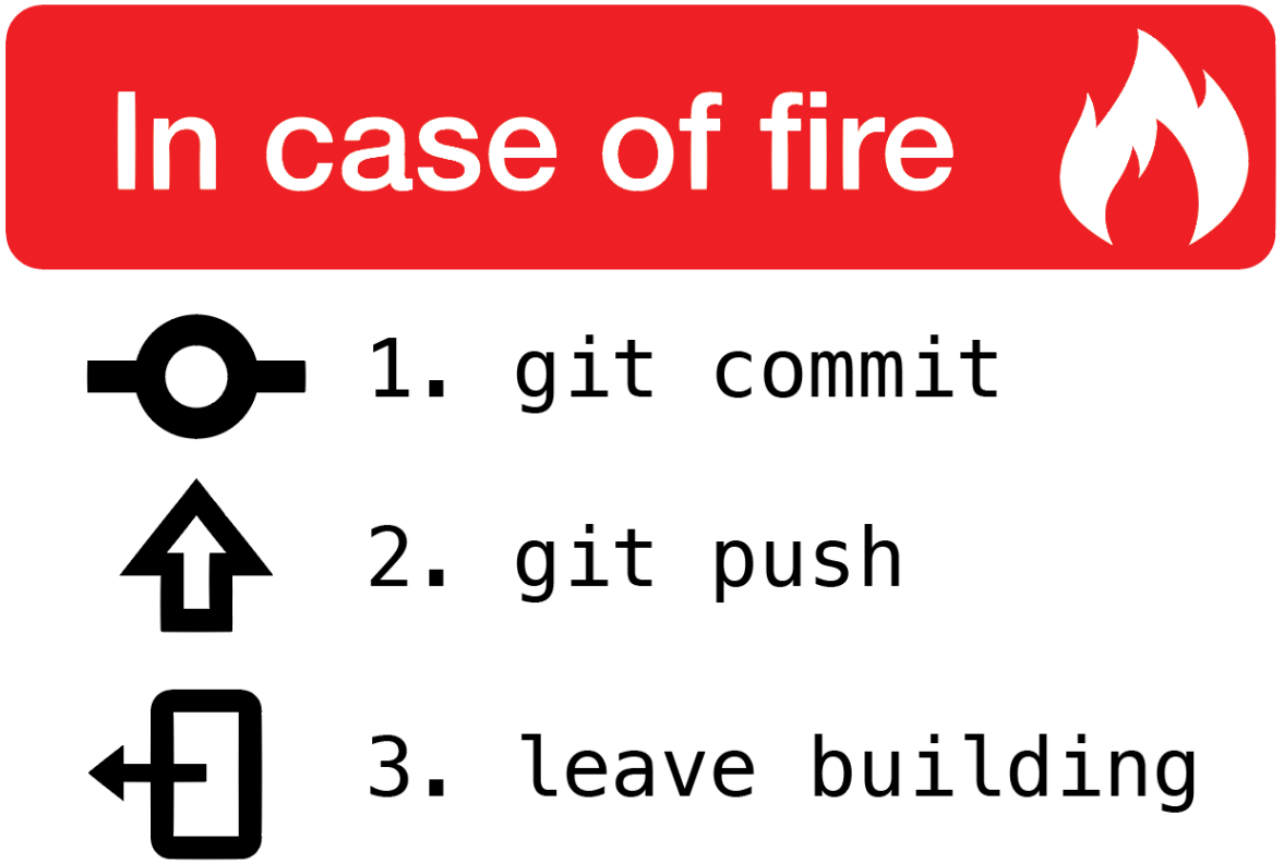


Adding a Remote Repo to Existing Project

```
 git remote add origin https://github.com/fxwalsh/msc-2019.git
 git remote -v
origin  https://github.com/fxwalsh/msc-2019.git(fetch)
origin  https://github.com/fxwalsh/msc-2019.git(push)
```

git push

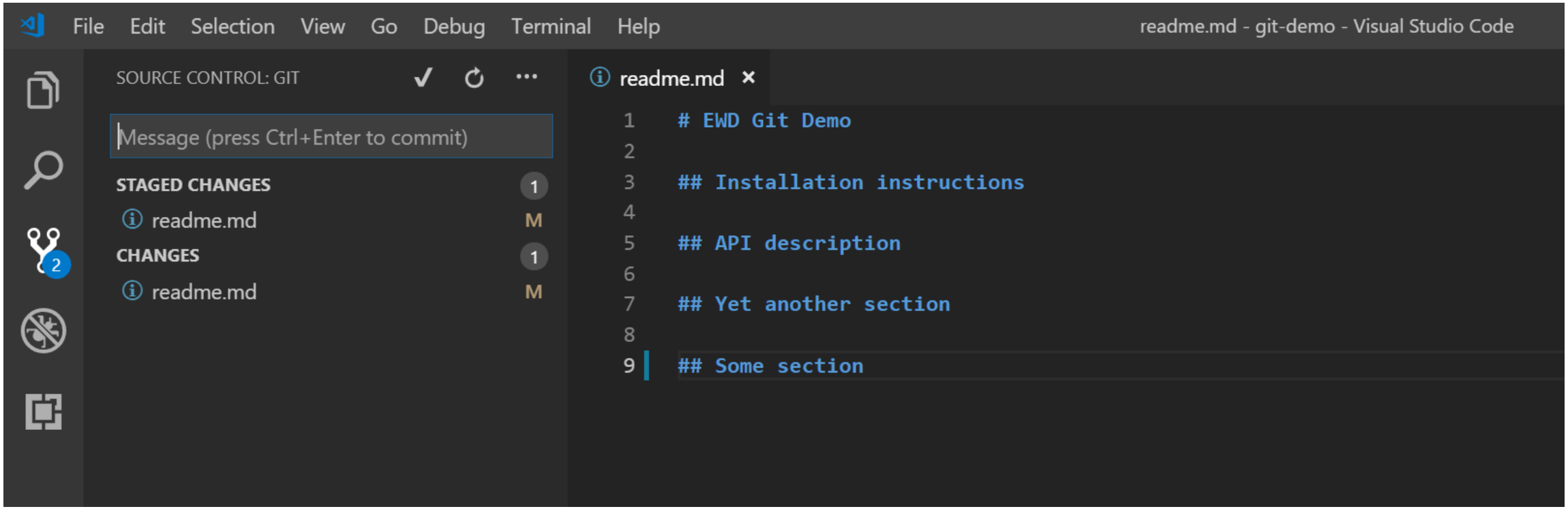
- Pushes your changes to remote
- Changes will be rejected if newer changes exist on remote
- Good idea to pull then push
  - merge locally, then push the results.



Pushing to remote

By default, remote repo is labelled **origin.**
To stage, commit, and push all your latest changes to origin…

```
git add --all
git commit –m “some important update to important stuff”
git push origin master
```


File Edit Selection View Go Debug Terminal Help
readme.md - git-demo - Visual Studio Code
SOURCE CONTROL: GIT
Message (press Ctrl+Enter to commit)
STAGED CHANGES 1
readme.md M
CHANGES 1
readme.md M
readme.md
1 # EWD Git Demo
2
3 ## Installation instructions
4
5 ## API description
6
7 ## Yet another section
8
9 ## Some section

Potential situation 1

I can't push to origin, it's saying "! [rejected] master -> master (fetch first)"

- Probably changes on the remote repo that you don't have locally.
- Solution: Do a **pull** then **push.**
- Should work as long and there's no conflict

```
$ git push
   To https://github.com/fxwalsh/project2.git
   ! [rejected]        master -> master (fetch
   first) error: failed to push some refs
….
$git pull
…..
   Merge made by the 'recursive' strategy.
   README.md | 2 +-
   1 file changed,1 insertion(+), 1 deletion(-)
$git push
Counting objects: 5, done.
…
```

Potential situation 2

I want to replace local repo with remote, don't care about losing changes

- Solution: fetch from the default remote, origin
- Reset your current branch (master) to origin's master

```
$ git fetch
…
$ git reset --hard origin/master
```

Potential situation 3

I want to replace remote repo with local, don't care about losing changes on remote

- Solution: force the push

```
$ git push -f origin master
```

Potential situation 4

Trying to pull but getting "CONFLICT (content): Merge conflict in ...."

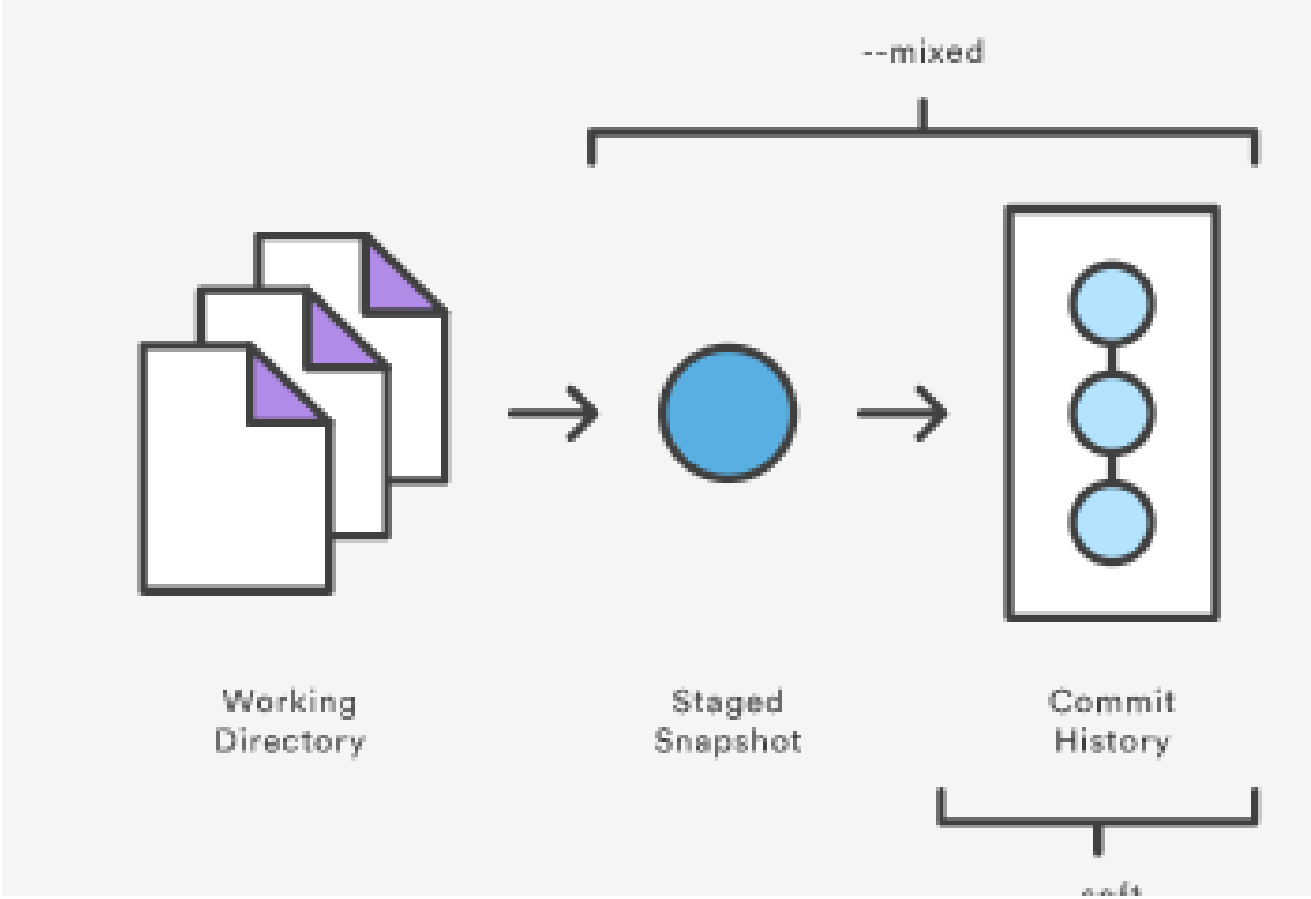
- Two different versions of the same file(s)
- Decide which one you want to keep manually.
- Perhaps force push or hard reset....

```
$ git push -f origin master
```

Potential Situation 5

I committed stuff I really didn't want to. I want to revert to the previous commit.

- git reset HEAD~1
  - Moves Head(pointer to latest Commit) back one
  - Leaves files alone (but not staged)
- git reset –hard HEAD~1
  - Nuclear option. Gets rid of changes completely (including working dir)
- git reset –soft HEAD~1
  - Leaves changes staged but resets head to previous commit



Branching and Merging

- Can have multiple local branches that can be entirely independent of each other.
- Allows:
- Role based codelines: (branch for production/testing/bug fixing)
- Disposable experimentation: Try something new – if it doesn't fly, delete the branch.
- Probably not vital when just 1 person/1 repo assignment.
- But if trying multiple approaches/features…